Optimal deterministic ring exploration with oblivious asynchronous robots

Anissa Lamani, Maria Potop-Butucaru, Sébastien Tixeuil

Optimal Deterministic Ring Exploration with Oblivious Asynchronous Robots

Anissa Lamani, Maria Potop-Butucaru, and Sébastien Tixeuil

Université Pierre et Marie Curie - Paris 6, LIP6-CNRS 7606, France

Abstract. We consider the problem of exploring an anonymous unoriented ring of size n by k identical, oblivious, asynchronous mobile robots, that are unable to communicate, yet have the ability to sense their environment and take decisions based on their local view. Previous works in this weak scenario prove that k must not divide n for a deterministic solution to exist. Also, it is known that the minimum number of robots (either deterministic or probabilistic) to explore a ring of size n is 4. An upper bound of 17 robots holds in the deterministic case while 4 probabilistic robots are sufficient. In this paper, we close the complexity gap in the deterministic setting, by proving that no deterministic exploration is feasible with less than five robots whenever the size of the ring is even, and that five robots are sufficient for any n that is coprime with five. Our protocol completes exploration in $O(n)$ robot moves, which is also optimal.

Key words: Robots, Anonymity, Obliviousness, Exploration, Asynchronous system, Ring

1 Introduction

Recent research focused on systems of autonomous mobile entities (that are hereafter referred to as *robots*) that have to collaborate in order to accomplish collective tasks. Two universes have been studied: the continuous euclidean space [8, 13, 4] where the robots entities can freely move on a plane, and the discrete universe in which space is partitioned into a finite number of locations, conventionally represented by a graph, where the nodes represent the possible locations that a robot can take and the edges the possibility for a robot to move from one location to the other [7, 11, 2, 1, 10, 9, 5, 6, 3]. In this paper we pursue research in the discrete universe and focus on the exploration problem when the network is an anonymous unoriented ring, using a team of autonomous mobile robots. The robots we consider are unable to communicate, however they can sense their environment and take decisions according to their local view. We assume anonymous and uniform robots (*i.e* they execute the same protocol and there is no way to distinguish between them using their appearance). In addition they are oblivious, *i.e* they do not remember their past actions. In this context, robots asynchronously operate in cycles of three phases: look, compute and move phases. In the first phase, robots observe their environment in order to get the position of all the other robots in the ring. In the second phase, they perform a local computation using the previously obtained view and decide on their target destination to which they will move in the last phase.

Related work In the discrete model, two main problems are investigated assuming very weak asynchronous, identical, and oblivious robots: the gathering and the exploration problem. In the gathering problem, robots have to gather in one location not known in advance *i.e* there exists an instant $t > 0$ where all robots share the same location (one node of the

ring). In the exploration problem, robots have to explore a given graph, every node of the graph must be visited by at least one robot and the protocol eventually terminates (that is, all robots are idle).

For the problem of gathering in the discrete robot model, the aforementioned weak assumptions have been introduced in [10]. The authors proved that the gathering problem is not feasible in some symmetric configurations and proposed a protocol based on breaking the symmetry of the system. By contrast in [9], they proposed a gathering protocol that exploits this symmetry for a large number of robots ($k > 18$) closing the open problem of characterizing symmetric situations on the ring which admit a gathering.

For the exploration problem, the fact that the robots have to stop after the exploration process implies that the robots somehow have to remember which part of the graph has been explored. Nevertheless, in this weak scenario, robots have no memory and thus are unable to remember the various steps taken before. In addition, they are unable to communicate explicitly, therefore the positions of the other robots remain the only way to distinguish different stages of the exploration process. The main complexity measure here is the minimal number of robots necessary in order to explore a given graph. It is clear that a single robot is not sufficient for the exploration in the case where it is not allowed to use labels. In [6], it has been shown that $\Omega(n)$ robots are necessary in order to explore trees of size n, however, when the maximum degree of the tree is equal to three then the exploration can be done with a sub-linear robot complexity. In the case where the graph is a ring, it has been shown in [5] that k (the number of robots) must not divide n (the size of the ring) to enable a deterministic solution. This implies that for a general n, $\log(n)$ robots are necessary. The authors also present in [5] a deterministic protocol using 17 robots for every n that is coprime with 17. By contrast, [3] presents a probabilistic exploration algorithm for a ring topology of size $n > 8$. Four probabilistic robots are proved optimal since the same paper shows that no protocol (probabilistic or deterministic) can explore a ring with three robots.

Contribution In this paper, we close the complexity gap in the deterministic setting. In more details, we prove that there exists no deterministic protocol that can explore an even sized ring with $k \leq 4$ robots. This impossibility result is written for the ATOM model [13] where robots execute their look, compute and move phases in an atomic manner, and thus extend naturally in the non-atomic CORDA model. We complement the result with a deterministic protocol using five robots and performing in the fully asynchronous non-atomic CORDA model [12] (provided that five and n are coprime). The total number of robot moves is upper bounded by $O(n)$, which is trivially optimal.

2 Model and Preliminaries

We consider a distributed system of mobile robots scattered on a ring of n nodes u_0,u_1,..., $u_{(n-1)}$ such as u_i is connected to both $u_{(i-1)}$ and $u_{(i+1)}$. The ring is assumed to be anonymous *i.e* there is no way to distinguish the nodes or the edges (i.e. there is no available labeling). In addition, the ring is unoriented *i.e* given two neighbors, it is impossible to determine which node is on the right or on the left of the other. On this ring k robots collaborate to

explore all the nodes of the ring. The robots are identical *i.e* they cannot be distinguished using their appearance and all of them execute the same protocol. Additionally, the robots are oblivious *i.e* they have no memory of their past actions. We assume the robots do not communicate in a explicit way. However, they have the ability to sense their environment and see the position of the other robots. Each robot can detect whether several robots are on the same node or not, this ability is called *multiplicity detection*. Robots operate in three phase cycles: Look, Compute and Move. During the Look phase robots take a snapshot of their environment. The collected information (position of the other robots) are used in the compute phase in which robots decide to move or to stay idle. In the last phase (move phase) they may move to one of their adjacent nodes towards the target destination computed in the previous phase.

At some time t, a subset of robots are activated by an abstract entity called *scheduler*. The scheduler can be seen as an external entity which selects some robots for the execution. In the following we assume that the scheduler is fair *i.e* each robot is activated infinitely many times. Two computational models exist: The *ATOM model* [13], in which synchronous cycles are executed in atomic way *i.e* the robots selected by the scheduler at the beginning of a cycle execute synchronously the full cycle, and the *CORDA model* [12] in which the scheduler is allowed to interleave different phases (For instance one robot can perform a look operation while another is moving). The model considered in our case is the CORDA model with the following constraint: the Move operation is instantaneous *i.e* when a robot takes a snapshot of its environment, it sees the other robots on nodes and not on edges. Nevertheless, since the scheduler is allowed to interleave the operations, a robot can move according to an outdated view (during the computation phase, some robots have moved).

In the following we assume that initially every node of the ring contains at most one robot. During the system execution a subset of robots are activated and move to other nodes. The position of all the robots at time t is the system configuration at t. During the Look phase, the activated robots take a snapshot of their environment in order to see the position of the other robots. The snapshot result is called a view and is defined by the two following sequences: $C^{+i}(t) =< d_i(t)d_{i+1}(t)...d_{i+n-1}(t) >$ and $C^{-i}(t) =< d_i(t)d_{i-1}(t)...d_{i-(n-1)}(t) >$ where $d_i(t)$ denotes the multiplicity of robots on the node u_i at instant t taking an arbitrarily orientation of the ring. $d_i > 1, \forall i \in [1, n]$ if and only if u_i is occupied by at least one robot, $d_i = 0$ otherwise. When $d_i(t) = 0$, the node u_i is said to be empty at instant t, when $d_i(t) = 1$, we say that the node u_i is occupied at instant t, otherwise we say that there is a *tower* on u_i at instant t.

The view at u_i is said to be *symmetric* at instant t if and only if $C^{+i}(t) = C^{-i}(t)$. Otherwise, the view of u_i is said to be *asymmetric*. When the configuration is symmetric, both edges incident to the node u_i which is occupied by the robot taking the snapshot look identical. In this case we assume the worst scenario allowing the adversary to take the decision on the direction to be taken.

Problem to be solved The problem considered here is the exploration problem, where k robots have to collaborate and explore a ring of size n before stopping forever. A protocol P solves the exploration problem if and only if the following conditions are satisfied:

1. **Safety** Every node is visited by at least one robot.
2. **Termination** The algorithm eventually stops.

3 Impossibility result

It has been shown [5] that no deterministic team of k robots can explore a ring of size n when k divides n. Also, it is known [3] that no exploration protocol (deterministic or probabilistic) is possible when $0 \leq k < 4$. We now observe that when a single robot is activated at a time, a trivial deterministic variation of the protocol of [3] (that uses four robots and randomization to break symmetry in some situations) matches the lower bound. So, our leveraging of the lower bound result of [3] to four robots considers the case when several robots can be activated at the same time.

Lemma 1 *There exists no deterministic protocol for exploring a ring of an even size (n) with four robots.*

Proof. The proof is by contradiction. We assume that there exists a deterministic protocol with four robots that can explore a ring and terminate. Then, we start from an admissible initial configuration where no two robots are located on the same node and derive executions that never satisfy the exploration specification.

We consider the two similar configurations shown in figures 1 and 2. As the configurations contain two axes of symmetry, the four robots $R1$, $R2$, $R3$ and $R4$ have identical views, which means that if they are activated simultaneously, they will exhibit the same behavior.

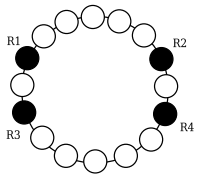

Fig. 1. Instance of configuration

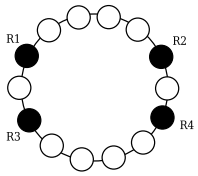

Fig. 2. Instance of configuration

1. *Suppose that every robot move towards it neighbor robot at distance* 2. Assume that all the robots are activated at the same time by the scheduler, then two towers are created (one with $R1$ and $R3$, the other with $R2$ and $R4$). From this point onwards, we assume that $R1$ and $R3$ are always activated simultaneously (and likewise for $R2$ and $R4$). As a result, the fours robots now behave as two robots. As it was shown in [3], no team of two robots can explore the ring, and thus the initial protocol does not perform a ring exploration either.
2. *Suppose that every robot move towards their adjacent node in the opposite direction of their neighbor robot at distance* 2. If the robots move back to their position, then the

protocol can never stop since the robots can go back and forth indefinitely. In the case where the robots keep moving away then two cases are possible :

– *The number of nodes between* $R1$ *and* $R2$ *is even* (the same for $R3$ and $R4$ see figure 2) in this case, by moving away from the robots that are at an odd distance from them, the configuration reached is similar to the one shown in figure 3 in which $R1$ and $R2$ are neighbors (the same for $R3$ and $R4$). Since the robots cannot go back (the protocol may never stop), the only move that they can perform is moving towards their neighbor: $R1$ moves towards $R2$ and vice versa (the same for $R3$ and $R4$), however, in the case where the four robots are activated at the same time, the two robots that are neighbors simply exchange their positions, and the configuration remains unchanged. As a result, no progress is made towards completion of the exploration task.

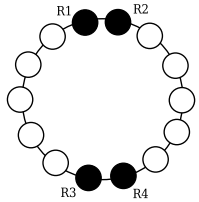

Fig. 3. Instance of configuration

– *The number of nodes between* $R1$ *and* $R2$ *is odd* (the same for $R3$ and $R4$, see figure 1). In this case k divides n, and [5] proved that the exploration problem in impossible to solve in this setting.

From the cases above, we can deduct that no deterministic exploration is possible using four robots when the size of the ring is even.

4 Ring Exploration in CORDA model

In this section we propose the ring exploration in Corda model with only five robots. Before detailing our algorithm we introduce some definitions. A **hole** is the maximal set of consecutive empty nodes, the size of a hole is the number of nodes that compose it, the border of the hole are the two empty nodes who are part of this hole, having one robot as a neighbor. An **inter-distance** d is the minimum distance taken among distances between each pair of distinct robots (in term of the number of edges). A **d.block** is any maximal elementary path in which there is one robot every d edges. The *border* of a d.block are the two robots belonging to this d.block having a hole of size bigger or equal to d as a neighbor. A **tower-chain** consists of an 1.block of size 3 followed by an empty node followed by a tower.

Our protocol consists of three distinct phases:

– **Block Module**. The aim of this phase is to drag all the robots in one single 1.block starting from any initial configuration that doesn't contain any tower.

– **Tower Module**. Starting from a configuration that contains a single 1.block, one tower is created in such way to give an orientation to the ring allowing the elected robot to explore the ring in the last phase.

– **Tower-chain Module** In this phase, starting from a configuration with a single tower, one robot is elected in order to explore the ring.

Algorithm 1 The orchestration of the algorithm

```
1: if the five robots do not form a tower-chain then
2:     if the configuration contains neither a tower nor a single 1.block then
3:         Execute Block Module
4:     else
5:         if the configuration contains a single 1.block then
6:             Execute Tower Module
7:         else
8:             Execute Tower-chain Module
9:         end if
10:    end if
11: end if
```

Note that once a configuration with a tower-chain is reached, the ring has been explored and the protocol terminates. Remark also that robots are able to distinguish the phase they are since each phase has different particularities. In the first phase all the configurations are tower-less and do not contain 1.block of size 5. In the second phase, configurations contain a single 1.block of size 5. And finally, in the last phase, the configurations contain a single tower.

The following section details and analysis the complexity of the previous modules.

Block Module description and analysis. The aim of this phase is to reach a configuration where there is a single 1.block that contains all the five robots without creating any tower. This phase is described in Algorithm 2.

Algorithm 2 Procedure: Block Module

```
1: if the configuration contains at least one isolated robot then
2:     if the configuration contains a single d.block then
3:         if I'm the isolated robot and I'm the closest neighbor to the d.block then
4:             Move toward the d.block taking the shortest hole
5:         end if
6:     else
7:         if the configuration contains two d.block then
8:             if the configuration is symmetric then
9:                 if I'm the isolated robot then
10:                    Move toward one of the two d.blocks
11:                end if
12:            else
13:                Move toward the closest d.block
14:            end if
15:        end if
16:    end if
17: else
18:    if the configuration contains a single d.block and d > 1 then
19:        if I'm at the border of the d.block then
20:            Move toward my adjacent node in the direction of the d.block
21:        end if
22:    else
23:        if the configuration contains two d.blocks then
24:            if I'm the smallest d.block and the closest to the biggest d.block then
25:                Move toward the biggest d.block
26:            end if
27:        end if
28:    end if
29: end if
```

Lemma 2 *If the configuration at instant t contains neither a single 1.block nor a tower, then the configuration at instant $t+1$ is tower-less.*

proof: We prove in this section that, if a robot moves, it moves always to an empty node to avoid the creation of towers. We suppose that the configuration at instant t is C. The configuration C doesn't contain any tower and satisfies one of the following cases:

– C contains at least one isolated robot: Two cases are possible according to the number of d.blocks:

1. There is a single d.block: in this case the isolated robots that are the closest to the d.block are allowed to move, they move to an empty node (see line 4). As it is an isolated robot, there are at least d empty nodes between it and the target d.block. In another hand, since $d \geq 1$, by moving, no tower is created at instant $t+1$.
2. There are two d.blocks: in this case, the configuration contains a single isolated robot (there are five robots on the ring), this robot is the only one allowed to move (see line $9, 10$), when it moves, it does to an empty node toward one of the two blocks depending of the symmetry of the configuration (there is no other robot between it and the target d.block and there are at least d empty nodes between it and the d.blocks – otherwise it would be part of them). Thus, no tower is created at instant $t+1$.

– C contains no isolated robots in the configuration: two cases are possible according to the number of d.blocks:

1. C contains a single d.block: in this case the robots at the border of this d.block are the only robots allowed to move, if they do, they move to an empty node toward the d.block they belong to. This guarantee is given by the condition $d > 1$ (see line 18). Hence, no tower is created at instant $t + 1$.
2. C contains two d.blocks: in this case the two d.blocks have different sizes (since there are no isolated robots and the number of robots is odd). Robots in the smallest d.block and closest to the biggest d.block move toward the target d.block taking the hole that separate them from one extremity of the biggest d.block. Since the size of the hole is at least equal to the inter-distance (otherwise robots are in the same d.block), no tower is created at instant $t + 1$.

Overall no tower is created at instant $t + 1$.

Lemma 3 *Starting from a configuration with no tower the system reaches a configuration with a single* 1*.block after* $O(n)$ *move operations.*

proof: Two cases are possible according to the type of the starting configuration denoted in the following C:

1. C contains at least one isolated robot: in this case, the robots allowed to move are always the isolated ones, and their destination is the closest d.block ($line 3, 4$), or one of the two d.blocks in the case of symmetry ($line 9, 10$). Hence three cases are possible according to the number of isolated robots:

   – The configuration C contains a single isolated robot. This robot is the only one allowed to move and its destination is the d.block. After its move the distance between it and the target d.block decreases. Since this robot remains the only isolated robot in the configuration (robots in the d.blocks do not move when there is at least one isolated robot), it is the only one that keeps moving to the same target d.block. Therefore, after a finite time, the robot joins the d.block.
   In order to compute the maximum number of moves (NBM) we consider the the worst case: the number of nodes between the two robots at the border of the d.block is odd. Consider the Figure 4.

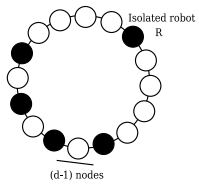

Fig. 4. Instance of configuration

Let compute the number of nodes between the d.block and the isolated robot. In order to do this, we will substract from the size of the ring, the occupied nodes and the empty nodes between the robots in the d.block.
The obtained result gives the sum of the empty nodes between the isolated robot and the d.block at each side. Thus, we have to divide it by two to obtain the distance between the isolated robot and the d.block. Note that the isolated robot is going to join the d.block hence it will advance until it reaches the same distance as the other robots in the d.block. Thus, in order to calculate the number of moves of the isolated robot, we have to substract from $n - [k + 3 * (d - 1)]$, the number of nodes between any two robots in the d.block.
The NBM is hence given by the following formula:

$$NBM = [\frac{n - [k + 3 * (d - 1)]}{2}] - (d - 1) \quad (1)$$

– The configuration contains two isolated robots: in this case, at least one of these two isolated robots is allowed to move. If there is a single robot that is the closest to the d.block, then this robot is the only one that moves, its destination is the single d.block (see line $3, 4$). At each move, the robot becomes even closer to the d.block, hence after a finite time, it reaches the d.block and the configuration contains a single isolated robot. In the other case (there are two robots allowed to move, let them be $R1$ and $R2$), whatever the choice of the scheduler in interleaving the different operations, at least one of these two isolated robots moves towards the d.block and hence becoming even closer. Note that if robots move at the same time both reduce their distance to the d.block.
Suppose the worst case: a single robot moves. Let $R1$ be this robot. Two cases are possible: If the robot that does not move, $R2$, has an up to date view of the configuration, then $R1$ becomes the closest robot to the d.block and hence it is the only one allowed to further move. From this point onward the proof is similar to the case 1. If $R2$ has an absolute view then $R2$ may also move. Consequently, either it is at the same distance as $R1$ from the d.block or $R1$ is the closest one to the d.block. The proof goes on recursively until at least one of the two robots reaches the d.block.
Note that the worst case happens when the two robots are at the maximum distance from the d.block and the number of empty nodes between these two robots is minimal (equal to d otherwise they form a d.block). It follows that the maximum number of moves robots perform in this case is given by the following formula:

$$NBM = n - (k + 4 * (d - 1) + d) \quad (2)$$

– The configuration contains three isolated robots. From the above cases above it follows that after a finite time all isolated robots join the d.block. The maximum number of moves in this case is performed when the three robots are at a maximum distance from the d.block and the distance between them is minimal and it is equal to d. This number is given by the following formula:

$$NBM = n - (k + 3 * (d - 1) + 2d) + [n - (k + 3 * (d - 1))]/2 - (d - 1) \quad (3)$$

Overall the number of isolated robots decreases until the configuration contains only d.blocks.

2. The configuration contains only d.blocks: Two sub-cases are possible according to the number of d.blocks:
   - The configuration contains two d.blocks. In this case the two d.blocks have different size and the smallest d.block moves towards the biggest one ($line 24, 25$). Consequently, whatever the choice of the scheduler at least one of the two robots moves towards the biggest one, when it does the configuration changes and contains a single d.block with isolated robots. However, it has been shown in case 1 that in this case and after a finite time all the isolated robots join the d.block. Hence the number of d.blocks decreases and the configuration contains a single d.block. The maximum number of moves is defined by the following formula and it happens when the small block is at a maximum distance from the biggest d.block:

$$NBM = n - (k + 5 * (d - 1)) \tag{4}$$

   - The configuration contains a single d.block: if $d > 1$ then there is at least one single node between each robot, and in this case, depending on the choice of the scheduler at least one of the two robots that are at the border of the d.block moves to its adjacent node in the direction of the d.block it belongs to ($line 19, 20$). Thus, the inter-distance decreases and the configuration reached contains isolated robots. However, once the configuration with a single d.block is reached, the maximum number of moves that are performed in order to reach another configuration with a single $(d-1)$.block $\forall d$ such as $d > 1$ is constant and is equal to 7. If one of the two robots at the border of the d.block moves, then one $(d-1)$.block is created and all the other robots move towards it starting with the closest one, since all the robots were at the same distance, the closest one performs one move to reach the $(d-1)$.block, the next robots performs two moves and the third one (the last one) performs three moves to reach the d.block. Hence, if we sum up all these moves taking in account the first move of the robot that creates the $(d-1)$.block, then the total number of displacements is the following: $1 + (1 + ... + k - 2)$ and is equal to 7. In the case where the two robots at the border of the d.block move at the same time, two $(d-1)$.block are created. The isolated robot that is on this axes of symmetry chooses one of them by moving towards it, when it moves it joins the chosen $(d-1)$.block and the configuration contains two $(d-1)$.blocks. However, in this case, only one robot is allowed to move (the closest one (see line $24, 25$), since the two robots in the smallest $(d-1)$.block are at different distance from the biggest one). This robot performs two moves to join the biggest $(d-1)$.block, the same for the second robot. Thus if we sum up the moves that were performed $(2 + 1 + 2 + 2)$, the total number is equal to 7. Since $d > 1$ the same process repeats the system reaches a configuration with a single 1.block. In this case the number of moves is given by the following formula:

$$NBM = (d - 1) * 7 \tag{5}$$

Since $d < n/5 - 1$ the total number of moves in order to reach a 1.*block* configuration starting from any tower-less configuration is $O(n)$.

Tower Module description and analysis. This phase begins when the configuration contains a single 1.block. It aims at creating a tower in order to give a virtual orientation to the ring such as the elected robot accomplish the exploration task in the last phase. This phase is described in Algorithm 3.

Algorithm 3 Procedure Tower Module

```
1: if I'm on the axes of symmetry then
2:     Move toward one of my neighbors
3: end if
```

Lemma 4 *Let C be the configuration that contains a single 1.block of size 5. If C is the configuration at instant t, then the configuration at instant $t+1$ contains a single tower.*

Tower-chain Module description and analysis. In this phase, one robot is elected in order to explore the ring. The exploration begins when the configuration contains a tower and is done when a tower-chain is created. This phase is described in Algorithm 4.

Algorithm 4 Procedure tower-chain Module

```
1: if the configuration doesn't contain a chain-tower then
2:     if I'm between the tower and the 1.block then
3:         Move toward my adjacent node in the opposite direction of the tower
4:     end if
5: end if
```

Lemma 5 *Starting from a configuration with a single tower, the system reaches a configuration that contains a chain-tower after $O(n)$ move operations and all the nodes have been explored.*

5 Conclusion

In this paper, we focused on the exploration problem in an undirected ring. We proved that no deterministic protocol can explore such a graph using k robots such as $k \leq 4$ if the ring is of even size. On the other hand, we provided a non-atomic completely asynchronous algorithm that uses only five robots for completing exploration provided that n and k are coprime. Our solution is thus optimal with respect to the number of robots. As exploration requires $O(n)$ robots moves, it is also optimal in time. We would like to mention two interesting open questions raised by our work:

1. The impossibility result of [5] shows that k must not divide n (for arbitrary values of k and n), while our impossibility result shows that k must be coprime with n (for a specific value of k: 4). We conjecture that the impossibility result of [5] can be extended to any k and n that are not coprime and such that $n > k$.

2. Our impossibility result holds for even sized rings only, thus remains the open question of designing a deterministic exploration protocol with fours robots for odd sized rings in the CORDA model (note that this is feasible in the ATOM model as presented in the appendix).

References

1. S. Das, P. Flocchini, S. Kutten, A. Nayak, and N. Santoro. Map construction of unknown graphs by multiple agents. *Theoretical Computer Science*, 385:34–48, 2007.
2. G. De Marco, L. Gargano, E. Kranakis, D. Krizanc, A. Pelc, and U. Vaccaro. Asynchronous deterministic rendezvous in graphs. *Theoretical Computer Science*, 355:315–326, 2006.
3. S. Devismes, F. Petit, and S. Tixeuil. Optimal probabilistic ring exploration by asynchronous oblivious robots. *In Proceedings of SIROCCO 2009*, 2009.
4. Y. Dieudonné, O. Labbani-Igbida, and F. Petit. Circle formation of weak mobile robots. *TAAS*, 3(16), 2008.
5. P. Flocchini, D. Ilcinkas, A. Pelc, and N. Santoro. Computing without communicating: Ring exploration by asynchronous oblivious robots. *OPODIS*, pages 105–118, 2007.
6. P. Flocchini, D. Ilcinkas, and N. Pelc, A.and Santoro. Remembering without memory: Tree exploration by asynchronous oblivious robots. *SIROCCO*, 5058:33–47, 2008.
7. P. Flocchini, E. Kranakis, D. Krizanc, N. Santoro, and C. Sawchuk. Multiple mobile agent rendezvous in a ring. *ISSN*, 2976, 2004.
8. P. Flocchini, P. Prencipe, N. Santoro, and P. Widmayer. Arbitrary pattern formation by asynchronous, anonymous, oblivious robots. *Theoretical Computer Science*, 407:412–447, 2008.
9. R. Klasing, A. Kosowski, and A. Navarra. Taking advantage of symmetries: Gathering of asynchronous oblivious robots on a ring. *OPODIS*, pages 446–462, 2008.
10. R. Klasing, E. Markou, and A. Pelc. Gathering asynchronous oblivious mobile robots in a ring. *Theoretical Computer Science*, 390:27–39, 2008.
11. D. Kowalski and A. Pelc. Polynomial deterministic rendezvous in arbitrary graphs. *ISAAC*, 3341:644–656, 2004.
12. G. Prencipe. CORDA: Distributed coordination of a set of autonomous mobile robots. In *Proc. 4th European Research Seminar on Advances in Distributed Systems (ERSADS'01)*, pages 185–190, Bertinoro, Italy, May 2001.
13. I. Suzuki and M. Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM Journal on Computing*, 28(4):1347–1363, 1999.

A Proof of Lemma 4

let $R1$ be the robot located in the middle of the 1.block in C. In C all robots execute algorithm 3 (see algorithm 1). Thus from C, $R1$ is the only one that can move, its destination is one of its adjacent node (see algorithm 3), by moving, a tower is created (the adjacent node was occupied) and the lemma holds.

B Proof of Lemma 5

When a tower is created as output of the Tower Module, there will be a single robot between the tower and the 1.block (let $R1$ be this robot). $R1$ is the only robot allowed to move, and if it does, it moves to its adjacent node in the opposite direction of the tower (see $line 2, 3$). After executing these actions, $R1$ remains the only robot between the tower and the 1.block. Consequently, it keeps moving to the same target destination (the 1.block). However, since at each move, the elected robot approaches the 1.block and since there are $(n-5)$ empty nodes between $R1$ and the 1.block at the beginning of tower-chain module, $R1$ will reach the 1.block and a chain-tower will be created after $n-5$ movements. Also, since the $n-5$ nodes that were between the isolated robot and the 1.block are the only nodes not explored (the five nodes that have been occupied in the beginning of phase 2 are already explored), when a configuration with a chain-tower is reached, all nodes of the ring have been explored and the protocol terminates (see algorithm 4 line 1).

A Exploration-Odd in ATOM model

In this section we propose a **deterministic** algorithm for the ring exploration problem using only **four** robots in the ATOM model. We consider rings of odd size n such that $n > 7$.

Before describing the algorithm, we first give some useful terms: A configuration contains an **S-tower-plan** if an only if the configuration is symmetric and the configuration contains two 1.blocks that share a hole of size 1 (see Figure 5). We say that a configuration contains an **A-tower-plan** if and only if the configuration is not symmetric and contains one 1.block of size 3 having an isolated robot as a neighbor at distance 2 (see Figure 6). A **tower-guide** is a path $u_i, u_{i+1}, u_{i+2}, u_{i+3}, u_{i+4}$, such as u_{i+2} and u_{i+4} are occupied by a single robot, u_i contains a tower, u_{i+1} and u_{i+3} are empty nodes. A configuration contains a **tower-block** if at each side of the tower there is one robot at distance 1 (see Figure 7). The configuration contains a **tower-sole** if the configuration is symmetric and there is an 1.block that does not contain the tower (see Figure 8).

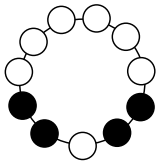

Fig. 5. S-tower-plan

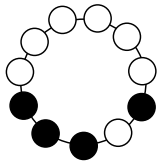

Fig. 6. A-tower-plan

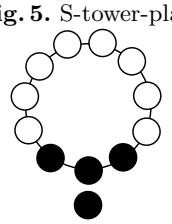

Fig. 7. Tower-block

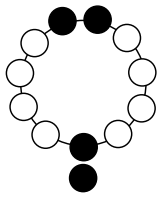

Fig. 8. Tower-Sole

A.1 Overview of the algorithm

Exploration-odd protocol consists of three phases:

- **Phase 1**: The aim of this phase is to reach a configuration with a S-tower-plan or an A-tower-plan within a finite time starting from any initial towerless configuration.

- **Phase 2**: Starting from a configuration that contains an (A or S)-tower-plan a tower is created. This tower is further used in the last phase.

- **Phase 3**: In this phase the exploration is performed by one or two robots according to the location of the tower. The protocol terminates when the configuration contains a tower-block or a tower-sole.

Algorithm 5 The protocol

```
1: if the configuration contains neither a tower-block nor a tower-sole then
2:     if the configuration contains neither a S-tower-plan nor an A-tower-plan then
3:         Execute Procedure Phase 1
4:     else
5:         if the configuration does not contain a tower then
6:             Execute Procedure Phase 2
7:         else
8:             Execute Procedure Phase 3
9:         end if
10:    end if
11: end if
```

A.2 Detailed description of the solution

In this section, we describe the different phases of the algorithm with more details, we provide an algorithm for each phase and we prove the correctness of our protocol.

Phase1 The aim of this phase is to reach a configuration with a (S or A)-tower-plan without creating any tower during the process. This phase is described in Algorithm 6. Let us define a couple of terms usefull in the sequel of the presentation. $Distance(X1, X2)$ is a function that returns the distance between the two robots $X1$ and $X2$. $distance(X)$ returns the distance between the robots satisfying property X. X=$RobotAxes$ refers to the two robots that are on the same side of the axes of symmetry, X=$SymRobotEv$ refers to the two symmetric robots that are at an even distance and X=$SymRobotOd$ refers to the two symmetric robots that are at an odd distance.

In the proofs below, in the case when the configuration is symmetric, we suppose that the two symmetric robots that are at an even distance from each other are $R1$ and $R2$, and the two other symmetric robots (that are at an odd distance) are $R3$ and $R4$. $R1$ and $R3$ are on the same side of the axes of symmetry (the same holds for $R2$ and $R4$). Recall that since the configuration is symmetric, $R1$ and $R2$ have identical views, which means that if they are activated simultaneously, they will exhibit the same behavior (the same for $R3$ and $R4$).

Algorithm 6 Procedure Phase 1

```
1: if the configuration is symmetric then
2:     if distance(symrobotEv) = distance(symrobotOd) + 1 then
3:         if distance(robotAxes) is even then
4:             if Distance(me, symRobot) is even then
5:                 if Distance(me, symRobot) > 2 then
6:                     Move towards my symmetric robot
7:                 else
8:                     Move in the opposite direction of my symmetric robot
9:                 end if
10:            end if
11:        else
12:            if distance(robotAxes)! = 1 then
13:                if Distance(me, symRobot) is odd then
14:                    Move in the opposite direction of my symmetric robot
15:                end if
16:            else
17:                if Distance(me, symRobot) is even then
18:                    Move towards my symmetric robot
19:                end if
20:            end if
21:        end if
22:    else
23:        if Distance(me, symRobot) is even and is bigger than 2 then
24:            if I'm in an 1.block then
25:                Move to my adjacent empty node towards my symmetric robot
26:            end if
27:        else
28:            if I'm not part of an 1.block that contains one robot that is not my symmetric robot then
29:                Move in the opposite direction of my symmetric robot
30:            end if
31:        end if
32:    end if
33: else/*the configuration is not symmetric*/
34:    if the configuration contains a single d.block of size 2 then
35:        if I'm the closest isolated robot then
36:            Move toward the d.block taking the shortest path
37:        end if
38:    else
39:        if the configuration contains a single d.block of size 3 then
40:            if d > 1 then
41:                if I'm at the border of the d.block then
42:                    Move in the direction of the d.block I belong to
43:                end if
44:            else
45:                if I'm the isolated robot then
46:                    if there is at least two nodes between me and the d.block on the shortest path then
47:                        Move towards the 1.block
48:                    end if
49:                end if
50:            end if
51:        end if
52:    end if
53: end if
```

Lemma 6 *If the configuration at instant t contains neither a tower nor a (S or A)-plan-tower then the configuration at instant $t+1$ is tower-less.*

Proof. Two cases are possible according to the type of the configuration

1. *the configuration is symmetric:* Two cases are possible:

– $Distance(R1, R2) = Distance(R3, R4) + 1$

- $Distance(RobotAxes)$ is even. In this case, $R1$ and $R2$ are the robots allowed to move. If $Distance(R1, R2) > 2$ then $R1$ and $R2$ move towards each other (see line $5, 6$), hence no tower is created (there are at least three empty nodes between them since $Distance(R1, R2) > 2$ and $R1$, $R2$ are at an odd distance). If $Distance(R1, R2) = 2$, as there are four robots and the size of the ring is bigger than 7, there are at least two empty nodes between the robots at each side of the axes of symmetry. Therefore no tower is created when $R1$ and $R2$ move towards $R3$ and $R4$ respectively (see line 8).
- $Distance(RobotAxes)$ is odd and is different from 1. In this case $R3$ and $R4$ are the two symmetric robots allowed to move. When they move, they go in the opposite direction of their symmetric robot (see line $13, 14$). Since the distance between the robots at the same side of the axes of symmetry ($R1$ and $R3$, the same for $R4$ and $R2$) is different from 1, then there is at least one empty node between each pair of robot at the same side of the axes of symmetry. Therefore, after the move is completed no tower is created.
- $Distance(RobotAxes)$. As there are four robots and the size of the ring is bigger than 7, there are at least three empty nodes between $R1$ and $R2$. Therefore, when robots move towards each other (see line $18, 19$) no tower is created.

– $Distance(R1, R2)$ is different from $Distance(R3, R4) + 1$. When $R1$ and $R2$ move (they are part of an 1.block), then there are at least 3 empty nodes between them (see line 23), thus,by moving no tower is created. When $R3$ and $R4$ move, we are sure that they are not in an 1.block that contains $R1$ and $R2$ respectively, thus there is at least one empty node between them and the robot at the same side of the axes of symmetry. Therefore, when robots move in the opposite direction of each other (see line $28, 29$) no tower is created.

2. *the configuration is not symmetric*: three cases are possible according to the type of the sub -configuration:

– *The configuration contains a single d.block of size* 2. In this case the closest isolated robot is allowed to move (there is only one since the configuration is not symmetric), when it does, it moves towards the d.block (see line $35, 36$). However, since it is an isolated robot, we are sure that there are at least d empty nodes between it and the d.block. On the other hand, since $d \geq 1$, then there is at least one empty node between the robot allowed to move and the target destination. Thus no tower is created.

– *The configuration contains a single d.block of size* 3. In this case, if $d > 1$, then the robots that are at the border of the d.block move in the direction of the block they belong to (see line $40, 41, 42$) and hence no tower is created (there is at least one empty node between robots at the border and the internal robot in the d.block since $d > 1$). In the case where

$d = 1$, the isolated robot is the one allowed to move if and only if it is not at distance 2 from the 1.block (see line $45, 46, 47$), hence there are at least two empty nodes between it and the 1.block. Thus when it moves, no tower is created.

Overall the configuration reached at instant $t + 1$ contains no tower.

Lemma 7 *Starting from a configuration that does not contain an axes of symmetry, a configuration with an A-tower-plan is reached in a finite time.*

Proof. According to the size of the d.block in the configuration, two cases are possible:

1. *the configuration contains a single d.block of size* 2. Since the configuration is not symmetric, there is exactly one robot which is the closest to the d.block. This robot is the only one allowed to move, its destination is the d.block (see line $(35, 36)$). By moving, the distance between it and the target d.block decreases. Hence after a finite time, it joins the d.block and the configuration contains one d.block of size 3 (we retrieve case 2).
2. *the configuration contains a single d.block of size* 3. If $d > 1$, then the robots that are at the border of the d.block move towards it (see line $(40, 41, 42)$). If the two robots are activated in the same time, then the configuration remains the same with a single (d-1).block of size 3. In the case where only one robot is activated, then a new d.block of size 2 is created with an inter-distance equal to $d-1$. In another hand, since the robot that was supposed to move is the closest one to the d.block, it is the only one allowed to move. After it completes the move it joins the new d-1.block, thus the configuration contains a single block of size 3 with an inter-distance equal to $d - 1$. Therefore after a finite time the configuration contains a single 1.block of size 3 and one isolated robot. The isolated robot is the only robot allowed to move while it not at distance 2 from the d.block (see line $(45, 46, 47)$). However, since at each move it becomes closer to the d.block, after a finite time, the isolated robot becomes neighbor of the 1.block at distance 2 and thus the configuration contains an A-plan-tower and the lemma holds.

Lemma 8 *Starting from a symmetric configuration in which distance*$(SymRobotEv) <$ *distance*$(SymRobotOd)$*, either the symmetry is broken or a configuration with an S-tower-plan is reached after a finite time.*

Proof. Two cases are possible according to the distance between the robots that are at the same side of the axes of symmetry.

– $Distance(RobotAxes)! = 1$. $R3$ and $R4$ are the only robots allowed to move, their destination is their adjacent node towards the robot being at the same side of the axes of symmetry. In the case where the scheduler activates only one of these two robots then the symmetry is broken and the lemma holds. Otherwise, the configuration remains symmetric and $Distance(R3, R4) > Distance(R1, R2)$. However, $distance(RobotAxes)$ decreases. Thus after a finite time, either the symmetry is broken or $Distance(R1, R3) = Distance(R2, R4) = 1$.

– $Distance(RobotAxes) = 1$. In the case where there is one empty node between the robots that are at an even distance ($R1$ and $R2$), then the configuration is terminal (S-tower-plan is created) and the lemma holds. Otherwise, $R1$ and $R2$ are the only robots allowed to move, their destination is their adjacent node towards each other. If the scheduler activates one of them then the symmetry is broken and the lemma holds. Otherwise, the configuration remains symmetric, $Distance(R1, R2) < Distance(R3, R4)$ and $Distance(R1, R3)! = 1$. In addition $Distance(R1, R2)$ decreases.

  Thus, in the case where the distance between the robots at the same side of the axes of symmetry is bigger than 1, the robots that are at an odd distance are the ones that moves, by moving distance(RobotAxes) decreases until it becomes equal to 1 (we suppose that the scheduler activates the symmetric robots at the same time, the symmetry is broken in the other case). When it is the case, the robots that are at an even distance are the one that move, by moving the distance between them decreases. Thus, starting from a symmetric configuration where $distance(RobotEv) < distance(RobotOd)$, either the symmetry is broken (in the case where the scheduler activates only one robot) or a terminal configuration is reached (configuration with an S-tower-plan).

Lemma 9 *Starting from a symmetric configuration in which* $distance(SymRobotEv) = distance(SymRobotOd) + 1$*, either the symmetry is broken or a configuration in which* $distance(SymRobotEv) < distance(SymRobotOd)$ *is reached in finite time.*

Proof. Three cases are possible:

1. $Distance(RobotAxes)$ is even. If $Distance(R1, R2) > 2$ then $R1$ and $R2$ are the robots allowed to move towards each other (see line 5, 6). In the case when one of them is activated by the scheduler (suppose that $R2$ is the one that moves), a new axes of symmetry is created, $R1$ and $R3$ become symmetric robots (the same for $R4$ and $R2$). In the new configuration the distance between $R4$ and $R2$ is bigger than the distance between $R1$ and $R3$ and the lemma holds. In the case, when both $R1$ and $R2$ are activated then by moving $Distance(R1, R2) < Distance(R3, R4)$ and the lemma holds.
   If $Distance(R1, R2) = 2$, then $R1$ and $R2$ remain the robots allowed to move. However, their destination changes (they move in the opposite direction of each other see line 8). In the case when the scheduler activated one of them the symmetry is broken and the lemma holds. Otherwise, $Distance(R1, R2) = Distance(R3, R4) + 3$. Since $Distance(R1, R2) > Distance(R3, R4)+1$, $R3$ and $R4$ are the only one allowed to move. If the scheduler activates only one of them then the symmetry is broken and the lemma holds. Otherwise, the new configuration verifies $Distance(R1, R2) = Distance(R3, R4) + 1$. However $Distance(R1, R2) > 2$, thus either the symmetry is broken or $Distance(R1, R2) < Distance(R3, R4)$. In both cases the lemma holds.
2. $Distance(RobotAxes)$ is odd and is different from 1. in this case the robots allowed to move are $R3$ and $R4$, their destination is the robot that is at the same side of the axes of symmetry ($R1$ and $R2$ respectively).In the case when they are activated at the same time, the configurations remains symmetric and $Distance(R3, R4) > Distance(R1, R2)$ (the lemma

holds). If the scheduler activates only one of them (let this robot be $R4$) then a new axes of symmetry is created however the distance between the robots that are at odd distance ($R3$ and $R1$) is bigger than the distance between the robots at an even distance ($R2$ and $R4$), hence the lemma holds.

3. $Distance(RobotAxes) = 1$. In this case the two robots $R1$ and $R2$ are the ones allowed to move (see line 17,18), their destination is their adjacent node in the direction of each other. In the case where the scheduler activates both of them at the same time then the configuration remains symmetric and the distance between them becomes smaller than $Distance(R3, R4)$. Thus the lemma holds. In the case where the scheduler activates only one of them then, a new axe of symmetry is created and distance(robotAxes) becomes odd. However, it has been shown in 2 that in this case either the symmetry is broken or $distance(SymRobotEv) < distance(SymRobotOd)$ and the lemma holds.

Lemma 10 *Starting from a symmetric configuration in which* $distance(SymRobotEv) > distance(SymRobotOd) + 1$*, then after a finite time, either the symmetry is broken or* $distance(SymRobotEv) = distance(SymRobotOd) + 1$.

Proof. Two cases are possible according to the distance of the robots at the same side of the axes of symmetry.

- $distance(RobotAxes)! = 1$. In this case, the two symmetric robots that are at an odd distance are the one allowed to move, their destination is their adjacent node in the direction of the robot that is at the same side of the axes of symmetry(see line 28,29). If the scheduler activates only one of them, then the symmetry is broken and the lemma holds. Otherwise, either the configuration remains the same however $distance(RobotAxes)$ decreases. or $Distance(R1, R2) = Distance(R3, R4) + 1$ or $Distance(R1, R2) > Distance(R3, R4) + 1$ and $distance(RobotAxes) = 1$. Hence after a finite time either the symmetry is broken or a configuration in which one of the two latter case is reached ($Distance(R1, R2) = Distance(R3, R4) + 1$ (the lemma holds) or $Distance(R1, R2) > Distance(R3, R4) + 1$ and $distance(RobotAxes) = 1$).
- $distance(RobotAxes) = 1$. $R1$ and $R2$ are the only robots allowed to move and they move towards each other. If the scheduler activates only one of them then the symmetry is broken and the lemma holds. In the other case, in the configuration reached either $Distance(R1, R2) = Distance(R3, R4) + 1$ and we are done or $Distance(R1, R2) > Distance(R3, R4)$ and $distance(RobotAxes)! = 1$. However the difference between distances that are between each pair of symmetric robots decreases.

From the two sub-cases above, we deduct that starting from a configuration in which $distance(SymRobotEv) > distance(SymRobotOd)$, either the symmetry is broken or a configuration in which $distance(SymRobotEv) = distance(SymRobotOd) + 1$ and the lemma holds.

Lemma 11 *Starting from any initial configuration that does not contain any tower, the system reaches a configuration with a (S or A)-plan-tower after a finite time.*

Proof. From lemma 7, 8, 9 and 10 imply that starting from any initial configuration that does not contain any tower, the system reaches a configuration with a (S or A)-plan-tower after a finite time.

Phase 2 This phase starts when the configuration contains a (S or A)-plan-tower. Its aim is to create a single tower in order to allow the exploration in the last phase. In the case where the configuration is symmetric the two robots that share a hole of size 1 move towards each other. If the configuration is not symmetric then the robot that is in the middle of 1.block moves towards its adjacent node not having a neighbor at distance 2. This phase is described in algorithm 7

Algorithm 7 Procedure Phase 2

```
1: if the configuration contains two 1.blocks then
2:     if I share a hole of size 1 with my symmetric robot then
3:         Move towards my symmetric robot
4:     end if
5: else
6:     if the configuration contains a single 1.block of size 3 then
7:         if I'm at in the middle of the 1.block then
8:             Move towards my adjacent node that doesn't have a neighbor at distance 2
9:         end if
10:    end if
11: end if
```

Lemma 12 *Starting from a configuration that contains a plan-tower, the system reaches a configuration with a single tower after a finite time.*

Proof. Two cases are possible according to the type of the plan-tower:

1. The configuration contains two 1.blocks. The robots allowed to move in such a configuration are the ones that share a hole of size 1 (see line 2, 3). In the case when they are activated at the same time by the scheduler, a single tower will be created on the axes of symmetry and the lemma holds. In the other case, the configuration reached contains one 1.block of size 3 having an isolated robot as a neighbor at distance 2, and we retrieve case 2
2. The configuration contains a single 1.block of size 3. In this case the robot that is in the middle of the 1.block moves towards its adjacent occupied node that has not a neighbor at distance 2. Hence a single tower is created and the lemma holds.

Phase 3 This phase is described in algorithm 8. In this phase the ring is explored using one or two robots according to location of the tower.

Algorithm 8 Procedure Phase 3

```
1: if there is no robot at distance 1 from the tower then
2:     if the configuration contains a tower-guide then
3:         if I'm not at distance 2 from the tower then
4:             if there is at least one empty node between me and the other isolated robot then
5:                 Move towards the isolated robot
6:             end if
7:         end if
8:     else if I'm at distance 2 from the tower then
9:         if I'm in an 1.block then
10:            Move towards the tower
11:        else
12:            Move in the opposite direction of the tower
13:        end if
14:    else
15:        if There is no robot at distance 2 from the tower then
16:            if I'm an isolated robot and I'm not at distance (n − 1)/2 from the tower then
17:                Move to my adjacent node in the opposite direction of the tower
18:            end if
19:        end if
20:    end if
21: else/*there is one robot at distance 1 from the tower*/
22:    if I'm not at distance 1 from the tower then
23:        Move towards the tower through the hole that we share
24:    end if
25: end if
```

Lemma 13 *Starting from a configuration with a tower, the system reaches a configuration that contains a tower-block or a tower-sole after a finite time and all the nodes have been explored.*

Proof. At the beginning of the phase, we are sure that at least one isolated robot is at distance 2 from the tower. If there is a single robot at distance 2 from the tower (let this robot be R1), then the tower was created from an A-tower-plan and the configuration contains a tower-guide. Suppose that the other isolated robot is $R2$. In such a configuration R2 is the only one that can move, when it is activated it moves towards $R1$ (see line $5, 6$) and a new 1.block is created. $R1$ is now, the only robot allowed to move, its destination is the tower (see line $10, 11$). When $R1$ becomes neighbor of the tower, we are sure that the nodes between $R1$ and $R2$ have been explored. $R2$ becomes the only robot that can move and this while there is a hole between it and the tower, its destination is the tower, hence , when it moves, it becomes a direct neighbor of the tower (at distance 1 from the tower). Since there was $n - 4$ nodes between the tower and $R2$ (all the nodes except the one that contains the tower, the nodes that are occupied by $R1$ and $R2$ and the empty node that was between $R1$ and $R2$ before moving $R2$), when $R2$ becomes neighbor of the tower, the configuration contains a tower-block and all the nodes have been explored. In the case where there are two robots at distance 2 from the tower (let them be $R1$ and $R2$), then the tower was created from a S-tower-plan and the two robots $R1$ and $R2$ are the only one allowed to move. Their destination is their adjacent node in the opposite direction of the tower (see line 12). In the

case where the scheduler activates only one of them, then the robot that is still at distance 2 from the tower is the only one that can move keeping the same destination (see line 13). After its move there will be no robots at distance 2 from the tower (the same case occurs when the two robots are activated at the same time by the scheduler). The two isolated robots are allowed to move until they become at distance $(n-1)/2$ from the tower and the configuration will contain a tower-sole. Thus, every robots explored $(n-5)/2$ nodes, *ie*, all the nodes have been explored.

Lemma 14 *Starting from any initial configuration, Odd-Exploration protocol terminates after a finite time.*

proof: Lemmas 11, 12 and 13 imply that, starting from any initial configuration that does not contain a tower, the system reaches a configuration with a tower-block or a tower-sole after a finite time. Note that all configurations with a tower sole or tower-block are terminal (see algorithm 5 line 1). Thus, Odd-Exploration protocol terminates after a finite time.